UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/851,554	05/08/2001	Stepan Sokolov	SUN1P833/P6212	4023

22434 7590 08/09/2005

BEYER WEAVER & THOMAS LLP
P.O. BOX 70250
OAKLAND, CA 94612-0250

EXAMINER	
YIGDALL, MICHAEL J	

ART UNIT	PAPER NUMBER
2192	

DATE MAILED: 08/09/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/851,554	SOKOLOV ET AL.	
	Examiner	Art Unit	
	Michael J. Yigdall	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *01 July 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4 and 6-21* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4 and 6-21* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on July 1, 2005 has been entered. Claims 1-4 and 6-21 are pending.

Response to Arguments

2. Applicant's arguments with respect to claims 1 and 9 have been considered but are moot in view of the new ground(s) of rejection necessitated by Applicant's amendment.

Claim Rejections - 35 USC § 101

3. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

4. Claims 1-4 and 6-21 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

With respect to claims 1-4 and 6-14, the language of the claims raises a question as to whether the claims are directed merely to abstract ideas that are not tied to a technological art, environment or machine which would result in a practical application producing a concrete, useful and tangible result to form the basis of statutory subject matter under 35 U.S.C. 101.

One simple remedy could be to replace the term "method" with --computer-implemented method-- in the preamble of each claim.

With respect to claims 15-19, the language of the claims raises a question as to whether the claims are necessarily implemented in hardware and instead directed merely to an arrangement of software *per se*, which is not tangible and therefore non-statutory.

With respect to claims 20 and 21, the language of the claims raises a question as to whether the claims are limited to tangible embodiments. Absent any explicit and deliberate definitions in the specification, the recited "computer readable medium" may include intangible embodiments. As such, the claims are not limited to statutory subject matter and are therefore non-statutory.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1-4, 6-8, 20 and 21 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pat. No. 5,903,899 to Steele, Jr. (art of record, "Steele") in view of U.S. Pat. No. 6,442,751 to Cocchi et al. (art now made of record, "Cocchi") in view of U.S. Pat. No. 6,304,949 to Houlsdworth (art of record, "Houlsdworth").

With respect to claim 1 (currently amended), Steele discloses a method of tracking references to objects of an object-oriented programming environment (see, for example, the abstract), said method comprising:

(a) determining whether a command is likely to place a reference to an object on an execution stack which is used to execute computer program code in said object-oriented programming environment (see, for example, column 8, lines 28-39, which shows determining whether the stack operands and results of an instruction or command are references, i.e. references to objects, and column 5, lines 62-65, which shows that the stack is used to execute computer program code).

Steele does not expressly disclose:

(b) determining whether there is a change in the flow control between the time said command is likely to place said reference to said object on said execution stack and the time said reference is used to access said object when said determining determines that said command is likely to place a reference to an object on an execution stack.

However, Cocchi teaches a method of tracking the types of local variables across subroutines (see, for example, column 4, lines 53-64), so as to precisely determine, for garbage collection, which local variables are references to heap-allocated objects (see, for example, column 1, lines 44-51). The local variables, and thus the references to the objects, are stored on a stack (see, for example, column 1, lines 14-16). Cocchi teaches identifying calls to the subroutines (see, for example, column 6, lines 6-18), or in other words, determining whether there are changes in the flow of control (see, for example, column 2, lines 45-48). The changes in the flow of control are between the time that the local variables are assigned and the time that

the local variables are accessed within the subroutines (see, for example, column 6, lines 1-5). Cocchi discloses that the method enables a garbage collector to identify dead objects and live objects, pointed to by references on the stack, even with the use of Java's JSR-subroutines (see, for example, column 9, lines 9-30) for which normal static bytecode analysis may not suffice (see, for example, column 2, lines 26-44).

Like Cocchi, Steele teaches locating the references stored on the stack for garbage collection in Java programs (see, for example, column 4, lines 23-29 and 62-66). One of ordinary skill in the art would have been motivated to improve the precision of Steele's garbage collection with the features taught by Cocchi, so as to identify references to objects even in the presence of Java's JSR-subroutines.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the method of Steele with the features taught by Cocchi, such that Steele determines whether there is a change in the flow control between the time said command is likely to place said reference to said object on said execution stack and the time said reference is used to access said object when said determining determines that said command is likely to place a reference to an object on an execution stack.

Steele also discloses:

(c) translating said command into another command when said determining determines that there is a change in the flow control (see, for example, column 8, lines 28-39, which shows replacing or translating the instruction into another instruction); and

(d) placing a reference to said object on a reference stack associated with said execution stack when said another command is executed (see, for example, column 7, lines 33-56, which

shows placing references on a reference stack during execution, and column 6, lines 57-65, which shows that reference stack is designated to store only references).

Steele does not expressly disclose the limitation wherein said reference stack is not used to execute computer program code and is designated to store only references to objects which have been stored in a heap.

However, Houlsdworth teaches a method of increasing the efficiency of garbage collection in a multi-threaded system (see, for example, column 1, lines 57-60). Houlsdworth discloses a reference stack that is separate from the stack frames used to execute computer program code (see, for example, column 5, lines 21-26) and is designated to store only references to objects stored in a heap (see, for example, column 4, lines 23-30).

Like Houlsdworth, Steele discloses multi-threaded systems (see, for example, column 6, lines 50-55). One of ordinary skill in the art would have been motivated to increase the efficiency of Steele's garbage collection in such multi-threaded systems with the features taught by Houlsdworth.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the method of Steele with the features taught by Houlsdworth, such that said reference stack is not used to execute computer program code and is designated to store only references to objects which have been stored in a heap.

With respect to claim 2 (previously presented), the rejection of claim 1 is incorporated, and Steele also discloses the limitation wherein said object-oriented programming environment is a Java compliant operating environment (see, for example, column 4, lines 62-66).

With respect to claim 3 (original), the rejection of claim 2 is incorporated, and Steele also discloses the limitation wherein said determining of whether a command is likely to place a reference on said execution stack is performed during Java Bytecode verification (see, for example, column 16, lines 10-18, which shows performing the determination when verifying Java bytecode).

With respect to claim 4 (original), the rejection of claim 3 is incorporated, and Steele also discloses the limitation wherein said determining of whether a command is likely to place a reference on said execution stack operates to determine whether a Getfield, Aload, Getstatic, or Return command is being performed (see, for example, column 13, lines 46-56, which shows identifying "getfield" and "getstatic" instructions for use with the reference stack, and column 7, line 65 to column 8, line 15, which shows the same for the "aload" instruction, and see, for example, column 8, lines 28-39, which shows identifying stack results, i.e. return values associated with a return command, as references).

With respect to claim 6 (previously presented), the rejection of claim 4 is incorporated, and Steele also discloses determining whether a "putfield" instruction is likely to place a reference to an object on the execution stack (see, for example, column 13, lines 46-56).

Houlsdworth further discloses:

(a) determining whether a Putfield command is likely to overwrite a reference to an object on the execution stack before said reference is used when said determining determines that there is not a change in the flow control (see, for example, column 6, lines 41-48, which shows detecting or determining whether a reference to an object on the execution stack is overwritten).

Steele also discloses:

(b) translating said command into another command when said determining determines that there is a Putfield command is likely to overwrite a reference to an object on the execution stack before said reference is used (see, for example, column 8, lines 28-39, which shows replacing or translating the instruction into another instruction).

With respect to claim 7 (original), the rejection of claim 1 is incorporated, and Houlsdworth further discloses the limitation wherein said reference stack and said execution stack have the same size (see, for example, column 5, lines 26-33, which shows that each reference stack is the same size as the stack frame, and column 4, lines 23-30, which shows that each stack frame in the execution stack has such a reference stack).

With respect to claim 8 (original), the rejection of claim 1 is incorporated, and Steele also discloses the limitation wherein at least one reference to an object is stored in an entry with an offset that is the same as the offset used to store said at least one reference in said execution stack, when said another command is executed (see, for example, column 14, lines 21-46, which shows that the references are stored in slots or entries and that the offsets are numbered such that the ordering of the offsets is the same as the original).

With respect to claim 20 (currently amended), the limitations recited in the claim are analogous to those of claim 1 (see the rejection of claim 1 above). Steele also discloses a computer readable medium including computer program code (see, for example, column 17, lines 20-27).

With respect to claim 21 (previously presented), the rejection of claim 20 is incorporated, and the limitations recited in the claim are analogous to those of claim 2 (see the rejection of claim 2 above).

7. Claims 9, 10, 15 and 16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Steele in view of Houlsdworth.

With respect to claim 9 (currently amended), Steele discloses a method of tracking references to Java objects in a Java programming environment (see, for example, the abstract and column 4, lines 62-66).

Steele discloses determining whether the stack operands and results of an instruction or command are references, i.e. references to objects (see, for example, column 8, lines 28-39, but does not expressly disclose:

(a) determining whether said Java command is likely to place the only reference to a Java object on the execution stack, wherein said reference is a direct reference to said Java object which has been stored in a heap designated for storing objects.

However, Houlsdworth teaches a method of increasing the efficiency of garbage collection in a multi-threaded system (see, for example, column 1, lines 57-60). Houlsdworth discloses determining whether a reference on the execution stack is the only reference to a Java object stored in a heap (see, for example, column 4, lines 12-19 and 23-30). The reference is a direct reference to the object (see, for example, column 4, lines 32-37).

Like Houlsdworth, Steele teaches garbage collection (see, for example, column 4, lines 23-29) and also discloses multi-threaded systems (see, for example, column 6, lines 50-55). One

of ordinary skill in the art would have been motivated to increase the efficiency of Steele's garbage collection in such multi-threaded systems with the features taught by Houlsdworth.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the method of Steele with the features taught by Houlsdworth, such that Steele determines whether said Java command is likely to place the only reference to a Java object on the execution stack, wherein said reference is a direct reference to said Java object which has been stored in a heap designated for storing objects.

Steele also discloses:

(b) translating said command into another command when said determining determines that said Java command is likely to place the only reference to a Java object on the execution stack (see, for example, column 8, lines 28-39, which shows replacing or translating the instruction into another instruction, and column 7, lines 57-64);

(c) executing said another Java command (note that the instructions or commands are inherently executed);

(d) placing a reference to said object on a reference stack associated with said execution stack when said another command is executed (see, for example, column 7, lines 33-56, which shows placing references on a reference stack during execution); and

(e) wherein said determining of whether said Java command is likely to place the only reference to a Java object on the execution stack is performed during Java Bytecode verification (see, for example, column 16, lines 10-18, which shows performing the determination when verifying Java bytecode).

With respect to claim 10 (original), the rejection of claim 9 is incorporated, and Steele also discloses the limitation wherein said determining that said Java command is likely to place the only reference to a Java object on the execution stack further comprises:

(a) determining whether a Getfield, Aload, Getstatic, or Areturn command is being performed (see, for example, column 13, lines 46-56, which shows identifying "getfield" and "getstatic" instructions for use with the reference stack, and column 7, line 65 to column 8, line 15, which shows the same for the "aload" instruction, and see, for example, column 8, lines 28-39, which shows identifying stack results, i.e. return values associated with an "areturn" command, as references).

With respect to claim 15 (currently amended), the limitations recited in the claim are analogous to those of claim 9 (see the rejection of claim 9 above). Steele also discloses a Java bytecode verifier (see, for example, column 16, lines 10-18).

With respect to claim 16 (original), the rejection of claim 15 is incorporated, and the limitations recited in the claim are analogous to those of claim 10 (see the rejection of claim 10 above).

8. Claims 11-14 and 17-19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Steele in view of Houlsdworth, as applied to claims 10 and 16 above, respectively, and further in view of Cocchi.

With respect to claim 11 (currently amended), the rejection of claim 10 is incorporated, but Steele does not expressly disclose the limitation wherein said determining that said Java

command is likely to place the only reference to a Java object on the execution stack further comprises:

(a) determining whether there is a change in the flow control.

However, Cocchi teaches a method of tracking the types of local variables across subroutines (see, for example, column 4, lines 53-64), so as to precisely determine, for garbage collection, which local variables are references to heap-allocated objects (see, for example, column 1, lines 44-51). The local variables, and thus the references to the objects, are stored on a stack (see, for example, column 1, lines 14-16). Cocchi teaches identifying calls to the subroutines (see, for example, column 6, lines 6-18), or in other words, determining whether there are changes in the flow of control (see, for example, column 2, lines 45-48). Cocchi discloses that the method enables a garbage collector to identify dead objects and live objects, pointed to by references on the stack, even with the use of Java's JSR-subroutines (see, for example, column 9, lines 9-30) for which normal static bytecode analysis may not suffice (see, for example, column 2, lines 26-44).

Like Cocchi, Steele teaches locating the references stored on the stack for garbage collection in Java programs (see, for example, column 4, lines 23-29 and 62-66). One of ordinary skill in the art would have been motivated to improve the precision of Steele's garbage collection with the features taught by Cocchi, so as to identify references to objects even in the presence of Java's JSR-subroutines.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the method of Steele with the features taught by Cocchi, such that Steele determines whether there is a change in the flow control.

With respect to claim 12 (original), the rejection of claim 11 is incorporated, and Steele also discloses determining whether a "putfield" instruction is likely to place a reference to an object on the execution stack (see, for example, column 13, lines 46-56).

Houlsdworth further discloses the limitation wherein said determining of whether said Java command is likely to place the only reference to a Java object on the execution stack further comprises:

(a) determining whether a Putfield command is likely to overwrite a reference to an object on the execution stack before said reference is used (see, for example, column 6, lines 41-48, which shows detecting or determining whether a reference to an object on the execution stack is overwritten).

With respect to claim 13 (original), the rejection of claim 12 is incorporated, and Houlsdworth further discloses the limitation wherein said reference stack and said execution stack have the same size (see, for example, column 5, lines 26-33, which shows that each reference stack is the same size as the stack frame, and column 4, lines 23-30, which shows that each stack frame in the execution stack has such a reference stack).

With respect to claim 14 (original), the rejection of claim 13 is incorporated, and Steele also discloses the limitation wherein at least one reference to a Java object is stored in an entry with an offset that is the same as the offset used to store said at least one reference in said execution stack, when said another command is executed (see, for example, column 14, lines 21-46, which shows that the references are stored in slots or entries and that the offsets are numbered such that the ordering of the offsets is the same as the original).

With respect to claim 17 (original), the rejection of claim 16 is incorporated, and the limitations recited in the claim are analogous to those of claim 11 (see the rejection of claim 11 above).

With respect to claim 18 (original), the rejection of claim 16 is incorporated, and the limitations recited in the claim are analogous to those of claim 12 (see the rejection of claim 12 above).

With respect to claim 19 (original), the rejection of claim 16 is incorporated, and the limitations recited in the claim are analogous to those of claims 17 and 18 (see the rejections of claims 17 and 18 above).

Conclusion

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is (571) 272-3707. The examiner can normally be reached on Monday through Friday from 7:30am to 4:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

MY

Michael J. Yigdall
Examiner
Art Unit 2192

mjy

ANTONY NGUYEN-BA
PRIMARY EXAMINER